CREATE DATABASE Компьютерный\_магазин;

GO

USE Компьютерный\_магазин;

GO

-- Создание таблицы "Товары"

CREATE TABLE Товары (

id\_товара INT PRIMARY KEY IDENTITY(1,1),

Название VARCHAR(100) NOT NULL,

Категория VARCHAR(50) NOT NULL,

Цена INT NOT NULL

);

GO

-- Создание таблицы "Покупатели"

CREATE TABLE Покупатели (

id\_покупателя INT PRIMARY KEY IDENTITY(1,1),

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Адрес VARCHAR(255) NOT NULL,

Телефон VARCHAR(20) NOT NULL

);

GO

USE Компьютерный\_магазин;

GO

-- Создание таблицы "Сотрудники"

CREATE TABLE Сотрудники (

id\_сотрудника INT PRIMARY KEY IDENTITY(1,1),

Имя VARCHAR(50) NOT NULL,

Фамилия VARCHAR(50) NOT NULL,

Должность VARCHAR(50) NOT NULL,

Зарплата INT NOT NULL

);

GO

-- Создание таблицы "Заказы"

CREATE TABLE Заказы (

id\_заказа INT PRIMARY KEY IDENTITY(1,1),

id\_покупателя INT,

id\_сотрудника INT,

Дата DATETIME NOT NULL,

FOREIGN KEY (id\_покупателя) REFERENCES Покупатели(id\_покупателя),

FOREIGN KEY (id\_сотрудника) REFERENCES Сотрудники(id\_сотрудника)

);

GO

-- Создание таблицы "Заказанные\_товары"

CREATE TABLE Заказанные\_товары (

id\_заказа INT,

id\_товара INT,

Количество INT NOT NULL,

FOREIGN KEY (id\_заказа) REFERENCES Заказы(id\_заказа),

FOREIGN KEY (id\_товара) REFERENCES Товары(id\_товара)

);

GO

-- Заполнение таблицы "Товары"

INSERT INTO Товары (Название, Категория, Цена)

VALUES ('Ноутбук Dell XPS 15', 'Ноутбуки', 100000),

('Монитор LG UltraGear 27GN950', 'Мониторы', 50000),

('Клавиатура Logitech G413', 'Клавиатуры', 5000),

('Мышь Razer DeathAdder V2', 'Мыши', 4000),

('Наушники Sony WH-1000XM4', 'Наушники', 30000);

-- Заполнение таблицы "Покупатели"

INSERT INTO Покупатели (Имя, Фамилия, Адрес, Телефон)

VALUES ('Анна', 'Смирнова', 'ул. Ленина 12', '1111111111'),

('Михаил', 'Кузнецов', 'ул. Победы 5', '2222222222'),

('Елена', 'Иванова', 'ул. Гагарина 7', '3333333333'),

('Сергей', 'Петров', 'ул. Кирова 3', '4444444444'),

('Ольга', 'Сидорова', 'ул. Пролетарская 9', '5555555555');

-- Заполнение таблицы "Сотрудники"

INSERT INTO Сотрудники (Имя, Фамилия, Должность, Зарплата)

VALUES ('Иван', 'Петров', 'Менеджер', 35000),

('Екатерина', 'Сидорова', 'Продавец', 30000),

('Алексей', 'Иванов', 'Менеджер', 35000),

('Мария', 'Петрова', 'Продавец', 30000),

('Дмитрий', 'Смирнов', 'Менеджер', 35000);

-- Заполнение таблицы "Заказы"

INSERT INTO Заказы (id\_покупателя, id\_сотрудника, Дата)

VALUES (1, 1, '2023-05-01 10:15:00'),

(2, 2, '2023-05-02 14:30:00'),

(3, 1, '2023-05-03 11:45:00'),

(4, 2, '2023-05-04 16:20:00'),

(5, 1, '2023-05-05 09:30:00');

-- Заполнение таблицы "Заказанные\_товары"

INSERT INTO Заказанные\_товары (id\_заказа, id\_товара, Количество)

VALUES (1, 1, 2),

(1, 3, 1),

(2, 2, 1),

(3, 4, 3),

(4, 5, 1),

(4, 6, 2),

(5, 3, 2),

(5, 6, 1);

CREATE VIEW Покупатели\_с\_заказами AS

SELECT Покупатели.Имя, Покупатели.Фамилия, COUNT(Заказы.id\_заказа) AS Количество\_заказов

FROM Покупатели

LEFT JOIN Заказы ON Покупатели.id\_покупателя = Заказы.id\_покупателя

GROUP BY Покупатели.Имя, Покупатели.Фамилия;

CREATE VIEW Сотрудники\_с\_высокой\_зарплатой AS

SELECT Имя, Фамилия, Должность, Зарплата

FROM Сотрудники

WHERE Зарплата > 30000;

CREATE VIEW Товары\_по\_категориям AS

SELECT Категория, COUNT(\*) AS Количество\_товаров

FROM Товары

GROUP BY Категория;

CREATE PROCEDURE ДобавитьТовар

@Название VARCHAR(100),

@Категория VARCHAR(50),

@Цена DECIMAL(10, 2)

AS

BEGIN

INSERT INTO Товары (Название, Категория, Цена)

VALUES (@Название, @Категория, @Цена)

END;

CREATE PROCEDURE ДобавитьПокупателя

@Имя VARCHAR(100),

@Фамилия VARCHAR(100),

@Адрес VARCHAR(255),

@Телефон VARCHAR(20)

AS

BEGIN

INSERT INTO Покупатели (Имя, Фамилия, Адрес, Телефон)

VALUES (@Имя, @Фамилия, @Адрес, @Телефон)

END;

CREATE PROCEDURE ПолучитьСписокЗаказов

AS

BEGIN

SELECT З.id\_заказа, П.Имя + ' ' + П.Фамилия AS Покупатель, С.Имя + ' ' + С.Фамилия AS Сотрудник, З.Дата

FROM Заказы З

INNER JOIN Покупатели П ON З.id\_покупателя = П.id\_покупателя

INNER JOIN Сотрудники С ON З.id\_сотрудника = С.id\_сотрудника

END;

CREATE TRIGGER ДобавлениеТовара

ON Товары

AFTER INSERT

AS

BEGIN

DECLARE @Название VARCHAR(100)

SELECT @Название = Название FROM inserted

PRINT 'Добавлен новый товар: ' + @Название

END;

CREATE TRIGGER ОтменаУдаленияЗаказа

ON Заказы

INSTEAD OF DELETE

AS

BEGIN

RAISERROR('Удаление заказа запрещено!', 16, 1);

END;

CREATE TRIGGER УдалениеСвязанныхЗаписей

ON Товары

AFTER DELETE

AS

BEGIN

DELETE FROM Заказанные\_товары

WHERE id\_товара IN (

SELECT id\_товара

FROM DELETED

);

END;

CREATE FUNCTION ПолучитьСписокТоваров()

RETURNS TABLE

AS

RETURN

(

SELECT id\_товара, Название, Категория, Цена

FROM Товары

);

CREATE FUNCTION ПолучитьСуммуЗаказа(@id\_заказа INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @сумма DECIMAL(10, 2);

SELECT @сумма = SUM(Товары.Цена \* Заказанные\_товары.Количество)

FROM Заказанные\_товары

INNER JOIN Товары ON Заказанные\_товары.id\_товара = Товары.id\_товара

WHERE Заказанные\_товары.id\_заказа = @id\_заказа;

RETURN @сумма;

END;

CREATE FUNCTION ПолучитьСписокСотрудниковПоДолжности(@должность VARCHAR(50))

RETURNS TABLE

AS

RETURN

(

SELECT id\_сотрудника, Имя, Фамилия, Должность, Зарплата

FROM Сотрудники

WHERE Должность = @должность

);

1. **Определите основные положения теории баз данных, хранилищ данных, баз знаний.**

База данных (БД) - организованная коллекция данных, которая хранится и управляется с использованием специальных программных средств.

Структура данных - описание способа организации данных в базе данных, включая определение таблиц, полей, связей и ограничений целостности.

Модель данных - абстрактное представление данных и их взаимосвязей в базе данных. Примеры моделей данных включают иерархическую, сетевую, реляционную и объектно-ориентированную модели.

Запросы - способы извлечения и модификации данных в базе данных. Запросы обычно формулируются с использованием языков запросов, таких как SQL.

Нормализация - процесс организации данных в базе данных с целью устранения избыточности и обеспечения эффективности работы с данными.

Хранилище данных — это специально организованная база данных, которая используется для агрегации, хранения и анализа больших объемов данных из различных источников. Основные положения хранилища данных включают:

1. Интеграция данных

2. Очистка данных

3. Исторические данные

База знаний — это система, которая хранит факты, правила, схемы и другую информацию, которая используется для поддержки принятия решений и решения проблем в определенной предметной области. Основные положения базы знаний включают:

1. Знания и факты

2. Принятие решений

3. Обновление и распределение знаний